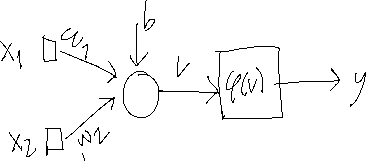
Ejercicio RRNN

Sea una viga biapoyada de sección bxh y longitud l, que está sometida a su peso propio, una carga distribuida de valor q y a una carga puntual en el centro del vano de valor P. Se pide definir una red neuronal que sea capaz de predecir la flecha en el centro del vano.

1. Esquema de la red



X1 = fuerza puntual.

X2 = carga distribuida.

b = sesgo (será la flecha debido a la carga muerta).

v = campo local inducido.

phi(v)= función de activación f(x)=x.

y = predicción.

Fase I - Entrenamiento

1. Se construye una base de datos para entrenar a la red a partir del modelo analítico de una viga biapoyada. Nos apoyamos de las fórmulas del prontuario (anexo A):

function [x,d] = datatrain(h,b,l,E,den,ntraining)

%

% Módulo de elasticidad hormigón E=30000; %N/mm2

% Ancho de la viga b=300; %mm

% Canto de la viga h=600; %mm

% Luz de la viga l=10000; %mm

% Densidad del material den=25; %N/mm3

% Numero de datos de entrenamiento ntraining=1000;

k1= l^3/(48\*E\*((b\*h^3)/12));

k2=(5\*l^4)/(384\*E\*((b\*h^3)/12));

cm=den\*b\*h\*10^-6;

fcm=cm\*k2;

%Genero una base de datos de flechas

for i=1:1:ntraining

f=rand()\*10000;

q=rand()\*10;

x(i,1)=f\*k1;

x(i,2)=q\*k2;

d(i,1)=f\*k1+q\*k2+fcm;

end

end

2. Inicializo los pesos y los sesgos a partir de los cuales voy a empezar mi entrenamiento

%inicializo los datos de w1,w2 y b

w1 = 0.1;

w2 = 1;

b = 5;

3. Defino la función coste: Como tenemos una regresión lineal es la función

4. Defino una función del algoritmo descenso gradiente

El algoritmo de actualización siguiendo el descenso gradiente es:

Para el caso concreto que nos ocupa de la función error es:

Sustituyendo en las ecuaciones anteriores obtenemos las ecuaciones para actualizar el descenso gradiente:

Fase II – Predicción.

Anexo A

Viga biapoyada con carga continua

Texto

Descripción generada automáticamente

Viga biapoyada con carga puntual en centro del vano

Imagen que contiene Interfaz de usuario gráfica

Descripción generada automáticamente